Sheet 1: Reverse Engineering and API Misuse Detection

Solutions to this sheet are due on Wednesday Oct. 29th, 23:59.

All assignments for this course are to be submitted through a version-control system called Subversion. We invite you to read Chapters 1 and 2 of the Subversion Book at `http://svnbook.red-bean.com/`.

For questions, we will be monitoring the discussion forum at `https://www.fachschaft.informatik.tu-darmstadt.de/forum//viewforum.php?f=555`. Please ask your questions there.

Exercise 1

a) Copy all necessary files from `public/sheet-1` into your svn folder (`svn cp public/sheet-1/ students/group-X/sheet-1`).

Change into the copied directory and then start Soot via command line:

```
java -jar soot.jar -allow-phantom-refs -process-dir ReverseMe.apk
-force-android-jar android.jar -src-prec apk -output-format jimple
```

NOTE: Better type the command-line instead of copy/pasting it, as otherwise there might be issues due to different character codes.

b) The command executed in exercise 1 a) has various Soot command line options. Explain in your own words the following Soot options:

- `-allow-phantom-refs`
- `-process-dir ReverseMe.apk`
- `-force-android-jar`
- `-src-prec apk`
- `-output-format jimple`

(Hint: Have a look into the "SootCommandLineOptions.pdf" in your project folder)

Furthermore, explain what the command in exercise 1 a) is supposed to do and describe the output files of Soot. (3 Points)

c) Now you should find a file called "de.ecspride.aca.sheet1.ReverseMe.jimple" in "sootOutput". Have a look into the "reverseMe1" method which implements malicious behavior. Explain in your own words what the code does. (1 Point)

d) Are there any automatic mechanisms that could detect this malicious method? If yes, explain the automatic mechanism. If no, explain why you think this is the case. (2 Points)

e) Have a look into the "reverseMe2" method, also in the ReverseMe.jimple. Reverse-engineer the method and explain in detail what it does. Further more, provide Java source code that corresponds to similar bytecode (jimple) output for the "reverseMe2" method. (2 Points)

Exercise 2

a) When the encryption algorithm DES was originally designed (in the 70s) a key length of 56-bit seems to be sufficient. But nowadays, modern computers are able to brute-force a 56-bit key and are thus able to break DES. So DES is considered as a weak cryptographic cipher and should not be used any more.

Import the Eclipse Project which can be found in `sheet-1/ICASheet1Template` to your Eclipse workspace. Complete the `BodyTransformer` in MainClass.java such that it reports statements of the form

```
Cipher.getInstance("DES");
```

which is a way of instantiating the DES cipher in Java. In the source folder `targets` you find bad code which your Transformer should be able to detect. Once the `BodyTransformer` found malicious code call `Reporter.report(SootMethod,Unit)` with the appropriate `SootMethod` and `Unit` to report the error. In the source folder `tests` you find some JUnit which your implementation should finally pass. (2 Points)

b) Get familiar with the Class `ConstantPropagatorAndFolder` and see how to extend the BodyTransformer from 2a) to spot if DES was used in this way. (1 Point)

```
String des = "DES";
Cipher.getInstance(des);
```

c) Give an example when the `ConstantPropagatorAndFolder` fails to find the misuse and explain why this happens. How could the constant folder be improved to also allow Soot to handle your example precisely? (2 Points)

d) Have a look inside the `TargetClass1.java` inside the `targets` source folder. The method `unreachable` is actually not called transitively from the `main` method, so it is unreachable. Figure out how to use a `SceneTransformer` such that the use of the DES cipher inside the `unreachble` method will not be reported, but the other misuse should still be reported. You can use the template class `ReachableMainClass`. (2 Points)

Submission: Hand in the solutions for exercises 1b)-e) and 2c) in a file called `solution.pdf`. Place this PDF file into your personal folder at `students/your-matrikelnummer/sheet-1`. For exercise 2 submit the whole Eclipse Project to this folder containing your implementation for `MainClass` and `ReachableMainClass`.

Hint: Tutorials for Soot can be found at `https://github.com/Sable/soot/wiki/Tutorials` and might be useful for solving these exercises.

Note on plagiarism

If you copy text elements from other sources, clearly mark those elements and state the source. Otherwise, we will have to consider your solution as a case of plagiarism, which will carry severe consequences such as failing this class.